# COMPUTERGESTÜTZTES EXPERIMENTIEREN II

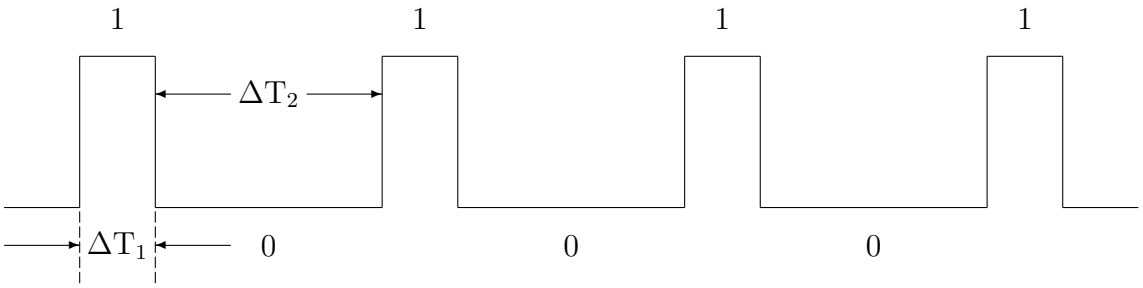
# P R A K T I K U M
# Programmieren in C/C++

## Blenden-Experiment

In diesem Experiment wird die Breite einer Schlitzblende mittels eines Schrittmotors und eines Schneckengetriebes eingestellt. Hinter der Blende befindet sich ein Lichtdetektor, der die Intensität des durch die Blende fallenden Lichtstrahles misst. Die Schnittstellenkarte USB-6009 von National Instruments deckt die gesamte Funktionalität ab, die für die Steuerung des Experimentes benötigt wird. Im ersten Teil lernen Sie, einen Schrittmotor mittels digitaler Signale zu steuern. Im zweiten Teil erfassen Sie mit einem Analog/Digital Umwandler ein analoges Signal, welches proportional der Lichtintensitt ist, und im dritten implemntieren Sie eine Regelung, die die Lichtintensitẗ auf einem vorgegebenen Wert hält.

## 1. Schrittmotorsteuerung

Der Schrittmotor soll mittels der digitalen Ausgabe der Schnittstellenkarte gesteuert werden. Hierzu werden die Bits 0 und 1 von Port0 benutzt. Das Bit0 legt die Drehrichtung fest (1 = Uhrzeigersinn, 0 = Gegenuhrzeigersinn = Blende öffnen). Durch Setzen und Rücksetzen des Bits1 werden Rechteckimpulse erzeugt, die jeweils einen Motorschritt bewirken:




Für Impulse gilt: $\Delta T_1 \ll \Delta T_2$

Die Zeit $\Delta T_1$ kann sehr kurz sein z.B. wenige $\mu$s. Es genügt die Zeit, die der Rechner benötigt, um ein Bit zu setzen und wieder zurückzusetzen. Die Zeit $\Delta T_2$ liegt im Bereich von 100 ms und kann mit mit der Prozedur void Sleep(unsigned long ms) realisiert werden.

Mit Hilfe des Schrittmotors wird die Schlitzblende geöffnet bzw. geschlossen. Zwei Schaltkontakte erfassen, ob die Blende völlig geschlossen oder völlig geöffnet ist. Die Schaltkontakte sind an Port1 angeschlossen, wobei Bit0 (LSB) = 1 des Datenregisters signalisiert, dass sich die Blende am oberen Anschlag befindet, - also völlig geöffnet ist -, und Bit1 = 1, dass die Blende völlig geschlossen ist.

Für den Betrieb des Schrittmotors wird empfohlen, folgende Prozeduren bzw. Funktionen zu erstellen:

| | |
|---|---|
| Konfig_Ports | Port0 und Port1 sollen wie im vorhergehenden Aufgabenblatt für die Ein-/Ausgabe konfiguriert werden |
| n=Schalter() | die Stellung der Endschalter, Bit0 und Bit1 von Port0, ist einzulesen. n: 1 Blende völlig geöffnet, 2 völlig geschlossen |
| Schritt_Zu() | der Motor soll einen Schritt im Uhrzeigersinn ausführen, aber nur wenn die Blende nicht völlig geschlossen ist |
| Schritt_Auf() | der Motor soll einen Schritt im Gegenuhrzeigersinn ausführen, aber nur wenn die Blende nicht völlig offen ist |

Das Programm soll in einer Borland C++ Form Applikation implementiert werden. Verwenden Sie LabelledEdit Fenster sowohl für die Eingabe der Zahl der auszuführenden Schritte N als auch fr die Anzeige der tatsächlich ausgegeführten Schritte und der relativen Position in Schritten. Steuern Sie über Bottons folgende Funktionen, wobei Sie die obigen Module verwenden

| | |
|---|---|
| ZU | Bewegen des Motors N Schritte im Uhrzeigersinn |
| AUF | Bewegen des Motors N Schritte im Gegenuhrzeigersinn |
| SCHLIESSEN | Schliessen der Blende. Diese Stellung kann als absolute Position (also Spatlbreite) genommen werden, wenn die relative Position zurück gesetzt wird |

Lernziele: Steuern eines Schrittmotors, Wartezeiten unterschiedlicher Grössenordnung, Zerlegen von Automationsaufgaben in elementare Steuerfunktionen.

# COMPUTERGESTÜTZTES EXPERIMENTIEREN II

# P R A K T I K U M
# Programmieren in C/C++

## Blenden-Experiment
## 2. Signalerfassung

Die Intensität des durch die Schlitzblende tretenden Lichtstrahles wird mit einem Lichtdetektor (Photowiderstand) gemessen. Der Detektor liefert eine Signalspannung zwischen 0 V und 5 V, die propotional zur einfallenden Lichtintensität ist und mit dem Analog/Digital-Konverter der USB-6009-Karte gemessen werden kann. Die Benutzung des A/D-Konverters ist in der separaten Anleitung für die USB-6009-Karte beschrieben (Appendix NIDAQmx).

Es ist eine Funktion zu schreiben, die die A/D-Umsetzung durchführt und folgendermassen aufgerufen wird:

volt = ADU()  volt: Wert der analogen Spannung

Das Signal ist an die Kanäle 0 und 1 des Multiplexers im Differenzspannungsverfahren angeschlossen. Zunächst muss der A/D-Konverter konfiguriert werden: Einstellen des Kanals0 und der Verstärkung (Spannungsbereich ±10 V) sowie Einstellen der differentiellen Spannungsmessung in einer Prozedur Konfig_ADU().

Erweitern Sie das Programm von Schrittmotorsteuerung I mit einem Button und einem Ausgabefenster, so dass die augenblickliche Spannung des Lichtdetektors gemessen und dargestellt werden kann.

Fügen Sie einen weiteren Button hinzu, der das Messen der Eichkurve

Spannung = f(Schrittzahl)

und deren grafische Darstellung mit TChart durchführt. Ablauf: zuerst Blende völlig schliessen; dann die Blende so weit öffnen, bis die Signalspannung ansteigt (ca. 400 Schritte), und anschliessend die Spannung als Funktion der Schrittzahl aufzuzeichnen (ca. 1500 Schritte).

Lernziele: Einlesen und Verarbeiten von analogen Signalen. Darstellung von Messkurven mit TChart.

# COMPUTERGESTÜTZTES EXPERIMENTIEREN II

# P R A K T I K U M
# Programmieren in C/C++

## Blenden-Experiment
## 3. Blendenregelung

Erweitern Sie das vorherige Programm um die Funktionalität einer Regelung, so dass die Spaltbreite der Blende kontinuierlich geregelt wird mit dem Ziel, eine vorgegebene Intensität des Lichtstrahles, der auf den Detektor fällt, aufrecht zu erhalten. Es handelt sich also um einen geschlossenen Regelkreis (direkte geschlossene Prozesskopplung):

D.h. wird die Intensität des Lichtstrahles, z.B. durch Abdunkeln der Lichtquelle, vermindert, ist die Schlitzblende nachzuregeln (d.h. zu öffnen), so dass die Intensität des auf den Detektor fallenden Lichts konstant bleibt.

Aufbauend auf dem Programm der vorhergenden Aufgabe soll zuerst die Kennlinie der Lichtintensität als Funktion der Spaltbreite gemessen und grafisch dargestellt werden. Der Sollwert soll dann mittels Grafikcursor eingegeben werden. Das Einstellen und Regeln soll ebenfalls grafisch dargestellt werden.

Da die Regelung in einer Endlosschleife implementiert wird, kann das GUI-Programm und damit die Regelung nicht unterbrochen werden. Abhilfe schafft die Ausführung der Regelung in einem separaten Thread. Hier lernen wir die ersten Schritte des Multithreadings: Erzeugen eines Threads (Instantiierung), Pausieren eines Threads (suspend) und weiteres Ausführen (resume).

Lernziel: Aufbau einer Regelschleife. Multithreading: Instatiierung, suspend, resume